一种改进的基于幂线性单元的激活函数 *

骆训浩，李培华

(大连理工大学 电子信息与电气工程学部，辽宁 大连 116024)

摘 要：非线性激活函数在卷积神经网络中扮演关键角色。针对修正线性单元（ReLU）完全丢弃网络中包含有用信息的负激活值问题，基于参数化修正线性单元（PReLU）和指数线性单元（ELU）的研究，提出一种新颖的参数化激活函数幂线性单元（PoLU）。PoLU 对输入的负激活部分实施有符号的幂非线性变化，幂函数的参数是可以在 CNNs 训练过程中自适应学习的；同时像 ReLU 那样保持正激活部分不变。PoLU 可以高效地实现并且灵活地运用到不同的卷积神经网络架构中。在广泛使用的 CIFAR-10/100 数据库上的实验结果表明，PoLU 要优于 ReLU 和它相对应的激活函数。

关键词：幂线性单元；参数化激活函数；卷积神经网络

中图分类号：TP183　　doi: 10.3969/j.issn.1001-3695.2018.04.0331

Improved activation function based on power linear unit

Luo Xunhao, Li Peihua

(*Faculty of Electronic Information & Electrical Engineering, Dalian University of Technology, Dalian Liaoning* 116024, *China*)

Abstract: The non-linear activation functions play an indispensable role in Convolutional Neural Networks (CNNs) . Aiming at the problem that ReLU completely discards negative activations which often contain much information. Based on the research of Parametric Rectified Linear Unit (PReLU) and Exponential Linear Unit (ELU) , this paper proposes a novel parametric activation function called Power Linear Unit (PoLU) . The proposed PoLU performs signed power non-linear transformation on negative activations. The parameters of power function can be learned adaptively during the training process of CNNs. Meanwhile, PoLU remain the positive activations unchanged. PoLU can be efficiently implemented and be flexibly adopted to various CNNs. The experimental results on widely-used CIFAR-10/100 benchmarks demonstrate PoLU are much better than ReLU and outperforms its counterparts.

Key words: power linear unit; parametric activation function; deep convolutional neural networks

1 激活函数

近年来，深度卷积神经网络（convolutional neural networks, CNNs）备受关注并且在许多计算机视觉任务中获得了引人注目的性能[8,17,18]。激活函数是 CNNs 中的基础单元，最近的研究表明，将修正线性单元（rectied linear unit, ReLU）作为激活函数是深度 CNNs 成功的关键[8,15,10,9]。ReLU 首次提出是用于限制玻尔兹曼机[7]，然后成功地用于神经网络[1]。ReLU 的定义是 $f(x)=\max\{0,x\}$。相比于传统的 Sigmoid 函数，ReLU 有两大优势[2]。首先，ReLU 可以缓解梯度消失问题[6]，同时可以加速收敛，并且可以避免网络收敛到一个局部最优解中。另外，ReLU 更趋向于得到稀疏的编码，这种稀疏的编码通常会带来更好的分类器性能[8]。

尽管 ReLU 可以为深度 CNNs 带来许多良好的性质，但是 ReLU 依然存在一些缺点。其中一个缺点是 ReLU 会忽略负激活，这些负激活通常会包含许多对表达目标有用的信息，尤其对于深度 CNNs 网络的浅层而言[3]。为了克服这个限制，许多改进 ReLU 的方法被提出来。表 1 对已有的方法做了总结。

泄漏的修正线性单元（leaky ReLU, LReLU）[2]对 ReLU 做了修改，LReLU 对负激活建模成一个线性函数，定义是

$$f(x)=\max\{0,x\}+a\min\{0,x\}$$

其中：$a=0.01$。LReLU 通过将负激活乘以一个数值小的标量，如 0.01，使得负激活可以在整个深度 CNNs 中传播。由于 LReLU 对于负激活有非零的导数，所以具有负激活值的参数也可以在端到端的学习中被更新。实验结果表明 LReLU 相比 ReLU 有更好的分类准确率。然而 LReLU 在整个网络中被人为地设置成

收稿日期：2018-04-08；**修回日期：**2018-05-24　**基金项目：**国家自然科学基金资助项目（61471082）

作者简介：骆训浩（1993-），男，湖北黄石人，硕士研究生，主要研究方向为图像分类、深度学习（luoxunhao@mail.dlut.edu.cn）；李培华（1973-），男，黑龙江人，教授，主要研究方向为计算机视觉、模式识别、统计机器学习.

相同的参数，这种是一种不合理的设置，因为负激活在深度 CNNs 的不同层中有不同的作用。为解决这个问题，He 等人[3]提出一种参数化的修正线性单元（parametric ReLU, PReLU）。这种激活函数在负激活部分引入一个带参数的线性变化，并且其参数可以与原始深度网络参数同时通过反向传播算法更新。PReLU 定义为

$$f(x)=\max\{0,x\}+a\min(0,x)$$

其中：a 是一个可以学习的参数。He 等人已经证实自动学习参数 a 要优于手工微调参数，如 LReLU。

不同于 LReLU 和 PReLU，另外一种最近提出来的激活函数是指数线性单元（exponential linear unit, ELU）[4]，它在负激活值中的表现为一种非线性变换：

$$f(x)=\max\{0,x\}+\min\{0,\beta(\exp(x)-1),0\}$$

其中：$\beta>0$。ELU 中的参数 β 通过手工来设定，通常设置为 1。

在 ELU 负激活处定义的非线性变换可以减小偏置变换，这使得标准梯度接近自然梯度，达到加速训练的目的。实验结果表明 ELU 在多种视觉任务上都优于其他激活函数。

ELU 表明在负激活处做非线性变换会优于线性变换。然而与 LReLU 相似，ELU 在深度 CNNs 所有层中，对负激活使用相同的非线性变换，这在实际场景中是不恰当的使用方式。基于以上的讨论并受到 PReLU 和 ELU 的启发，本文提出一种新颖的参数化的激活函数——幂线性单元（power linear unit, PoLU）。如表 1 和图 1 所示，不同于现有激活函数，通过引入一个可学习的参数，本文提出的 PoLU 激活函数可以在不同的深度 CNNs 层呈现出不同的形式。另外，PoLU 可以有效地实现并被灵活地使用到现有的深度 CNNs 网络中。本文实验在广泛使用的数据集上进行：CIFAR-10 和 CIFAR-100。实验结果表明 PoLU 在深度卷积神经网络架构 Network in Network[10]上都要优于其他相应的激活函数。

表 1　现有激活函数的对比

方法	$f(x)\{x>0,x\le0\}$	$\frac{\partial f(x)}{\partial x}\{x>0,x\le0\}$	$\frac{\partial f(x)}{\partial a}\{x>0,x\le0\}$
ReLU[1]	$\{x,0\}$	$\{1,0\}$	-
LReLU[2]	$\{x,0.01x\}$	$\{1,0.01\}$	-
PReLU[3]	$\{x,ax\}$	$\{1,a\}$	$\{0,x\}$
ELU[4]	$\{x,\beta(\exp(x)-1)\}$	$\{1,\beta\exp(x)\}$	-
PoLU(本文)	$\{x,-\lvert x\rvert^{a}\}$	$\{1,-a\lvert x+\varepsilon\rvert^{a-1}\}$	$\{0,\lvert x\rvert^{a}\ln\lvert x+\varepsilon\rvert\}$

注: $f(x)$、$\frac{\partial f(x)}{\partial x}$ 和 $\frac{\partial f(x)}{\partial a}$ 分别表示前向传播、关于输入 x 的反向传播和关于参数 a 的反向传播。本文的 PoLU 有一个额外的正则化系数 ε

2　幂线性单元

本章详细说明本文提出的幂线性单元（PoLU）。两种类型的 PoLU 被提出：通道共享型（channel-shared case）和通道独享型（channel-wise case）。

2.1　前向传播

通道独享型 PoLU 定义如下：

$$f(x_i)=\begin{cases}x_i & if \quad x_i>0\\ -\lvert x_i\rvert^{a_i} & if \quad x_i\le 0\end{cases} \tag{1}$$

其中：x_i 表示非线性激活函数 $f(\cdot)$ 在第 i 个通道的输入；a_i 是一个可学习的参数，可以用来控制负激活部分的非线性，在这里限制学习参数 $a_i>0$。

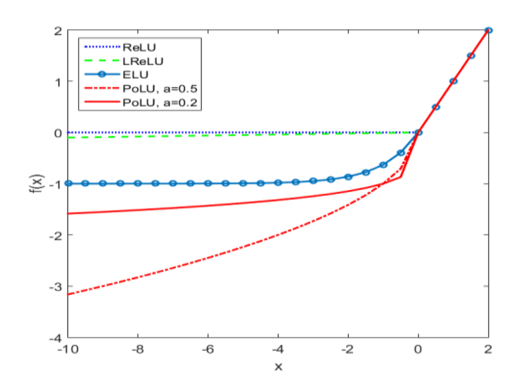


图 1　修正线性单元(ReLU),泄漏修正线性单元(LReLU, $a=0.01$),指数线性单元(ELU, $a=1.0$),幂线性单元(PoLU, $a=0.5$ 和 $a=0.2$)

下标 i 表示第 i^{th} 个通道，因此对于不同的通道可以学习到不同的参数 a_i，各个通道都有自己的非线性表现。当 $a_i=0.5$ 时，PoLU 表现为平方根。a_i 越小，所得到的负激活值越小，负激活部分越靠近 x 轴；反之，负激活部分越远离 x 轴。PoLU 通过这个特性来控制负激活部分的非线性表现。图 1 显示了 $a=0.5$ 和 $a=0.2$ 时 PoLU 和其他激活函数的比较。利用可学习参数 a_i，深度 CNNs 可以在网络训练阶段控制它的非线性表现。根据式（1）中 PoLU 的定义，越小的 a_i 会带来更多非线性表现。

2.2　反向传播

PoLU 可以使用反向传播算法训练参数。根据链式法则，损失函数 E 关于参数 a_i 的导数是：

$$\frac{\partial E}{\partial a_i}=\sum_{x_i}\frac{\partial E}{\partial f(x_i)}\frac{\partial f(x_i)}{\partial a_i} \tag{2}$$

其中：$\frac{\partial E}{\partial f(x_i)}$ 是从网络深层传播过来的梯度。

与此同时，$\frac{\partial f(x_i)}{\partial a_i}$ 是激活函数 $f(\cdot)$ 关于 a_i 的导数：

$$\frac{\partial f(x_i)}{\partial a_i}=\begin{cases}0 & if \quad x_i>0\\ \lvert x_i\rvert^{a_i}\ln\lvert x_i+\varepsilon\rvert & if \quad x_i\le 0\end{cases} \tag{3}$$

其中：ε 是一个足够小的正实数，本文中设为 $1e-8$；求和项 $\sum_{x_i}$ 表示对输出的特征映射一个通道的全部位置元素求和。另外，损失函数 E 关于输入 x_i 的导数如下：

$$\frac{\partial E}{\partial x_i}=\frac{\partial E}{\partial f(x_i)}\frac{\partial f(x_i)}{\partial x_i} \tag{4}$$

式（5）中，$\frac{\partial f(x_i)}{\partial x_i}$ 是激活函数 $f(\cdot)$ 关于 x_i 的导数：

$$\frac{\partial f(x_i)}{\partial x_i}=\begin{cases}1 & if \quad x_i>0\\ -a_i\left|x_i+\varepsilon\right|^{a_i-1} & if \quad x_i\le 0\end{cases} \tag{5}$$

参数的更新规则如下：

$$\Delta a_i \leftarrow \mu\cdot\Delta a_i+\omega\cdot\alpha\cdot a_i+\alpha\cdot\frac{\partial E}{\partial a_i} \\ a_i\leftarrow a_i+\Delta a_i \tag{6}$$

$$\Delta x_i \leftarrow \mu\cdot\Delta x_i+\omega\cdot\alpha\cdot x_i+\alpha\cdot\frac{\partial E}{\partial x_i} \\ x_i\leftarrow x_i+\Delta x_i \tag{7}$$

式（6）（7）中：μ 是动量系数；α 是学习率；ω 是权重衰减系数。

2.3 通道共享型

通道独享类型 PoLU 某一层的所有通道参数 a 都不相同，其额外参数数量等于相应网络层的通道数量。相反，通道共享型 PoLU 的参数 a 被一个网络层的所有通道共享，每一个 PoLU 单元只引入一个额外参数。通道共享型 PoLU 的反向传播如下所示：

$$\frac{\partial E}{\partial a}=\sum_i\sum_{x_i}\frac{\partial E}{\partial f(x_i)}\frac{\partial f(x_i)}{\partial a} \tag{8}$$

$$\frac{\partial f(x_i)}{\partial a}=\begin{cases}0 & if \quad x_i>0\\ \left|x_i\right|^a \ln\left|x_i+\varepsilon\right| & if \quad x_i\le 0\end{cases} \tag{9}$$

式（8）中：$\sum_i$ 表示深度 CNNs 某一层中所有通道的梯度和。这种通道共享型 PoLU 每一层只引入一个额外的参数 a，额外的内存消耗和计算代价可以忽略不计。相比通道独享型 PoLU，通道共享型 PoLU 引入的参数更少，计算效率更高。但是正如实验所示，通道独享型 PoLU 有更好的性能。

3 实验结果及分析

本章详述实验部分。本文在典型的深度 CNNs 网络架构 network in network（NIN）上进行实验。本文与四种不同的激活函数进行比较，这四种激活分别是 ReLU[1]、LReLU[2]、PReLU[3]和 ELU[4]。比较实验在 CIFAR-10、CIFAR-100 数据库进行。

3.1 实验数据库

CIFAR-10（C10）数据库由 10 个类别的彩色自然场景图像组成，包含 50 000 幅训练图像和 10 000 幅测试图像，每幅图像是大小为 32×32 的 RGB 图像。本文使用文献[5]所示的图像预处理方式，即颜色归一化和 ZCA 白化。训练阶段的数据增广的方式与文献[10,13,11]一致，即在原始图像的每一侧都填充 4 个像素，变成 40×40 大小的填充图像，并在填充图像中随机裁剪出 32×32 大小的图像，以 0.5 的概率随机水平翻转裁剪图像。在测试阶段只使用颜色归一化的图像。与文献[11]一致，采用这种数据增广方式的数据库叫做 C10+。CIFAR-100（C100）数据库与 C10 相同，区别是 C100 包含 100 个类别的图像。C100+的数据增广方式与 C10+相同。

3.2 在 NIN 中的实验结果

本文使用的 NIN 网络结构与文献[10]一致，由三个级联的 mlpconv 层组成，每一个 mlpconv 层后面都接着一个空间池化层用作下采样。每个 mlpconv 层包含一个卷积层和两个级联跨通道参数池化（cascaded cross channel parametric pooling，cccp）层组成。其中 cccp 层等价于卷积核是 1×1 的卷积层。最后一个 mlpconv 层只包含一个卷积层和一个 cccp 层。除了最后一个 mlpconv 层外，其他层都使用了 Dropout[14] 技术。NIN 的最后一层是一个全局平均值池化层，一个 k 通道的全连接层和一个 softmax 层。

带有 PoLU 的网络的训练流程与典型的 AlexNet[8]训练流程相似。网络使用随机梯度下降算法训练，批处理大小是 128，动量系数是 0.9，权重衰减系数是 0.000 5。初始化的学习率是 0.04，训练 80 个 epoch 后当验证错误率趋于稳定时，学习率降低 10 倍。

3.2.1 手工设置参数 a 实验分析

PoLU 激活函数中有一个关键参数 a，PoLU 通过这个参数来控制负激活部分的非线性表现。本节实验将验证参数 a 对 PoLU 的性能的影响，实验所用的 PoLU 类型是通道共享型。这样 PoLU 的每一层都共享同一个参数，便于排除其他因素的干扰。实验所用的数据库是 C10，其 baseline 的测试误差是 10.41%[10]。在对比实验中，本文给 NIN 中每个 PoLU 层的参数 a 都手工设置为相同的数值，总共进行 5 组对比实验，实验评价指标是网络 Top-1 测试误差（%）。表 2 是对比实验结果。其表明当手工设置参数 a 来控制网络 PoLU 层的非线性表现时，参数值过大或者过小都会出现性能的损失。并且在对比实验中，参数 a 被设置成所有 PoLU 层数值相同，这意味着所有 PoLU 层的非线性表现相同。这种设置是不合理的，因 CNN 的每一个卷积层都有不同的非线性，其对应的 PoLU 层参数也都各不相同。如果手工地去设置每一个 PoLU 层的参数 a，这将是很大的工作量，并且需要积累很多经验。本文将参数 a 引入为一个可学习的参数，在训练 CNN 的过程自适应地更新，CNN 的每一个卷积层对应的 PoLU 层可以学习到本层最适应的参数值。

表 2 参数 a 对 PoLU 的影响对比实验(%)

a	0.3	0.4	0.5	0.6	0.7
Top-1	9.68	9.20	9.27	9.23	9.56

3.2.2 通道共享型和通道独享型非线性比较

首先，在 C10 上训练一个带有 ReLU 激活函数的 NIN 作为 baseline，训练好的网络 top-1 错误率是 10.10%。接着，NIN 中所有的 ReLU 激活函数被替换成 PoLU 激活函数，从头开始训练这个网络，使用的设置与将 ReLU 作为激活函数的 NIN 一致。NIN 中的 PoLU 层是通道独享型，其中参数 a 的初始化设为 0.5。图 2 和 3 分别这两个网络的训练误差和测试误差。为了更清楚地显示测试误差的收敛趋势，图 4 展示了第 60 个 epoch 之后的测试误差曲线。

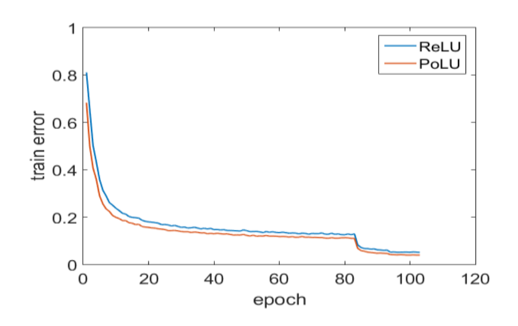


图 2　NIN 使用 ReLU 和 PoLU 的训练误差

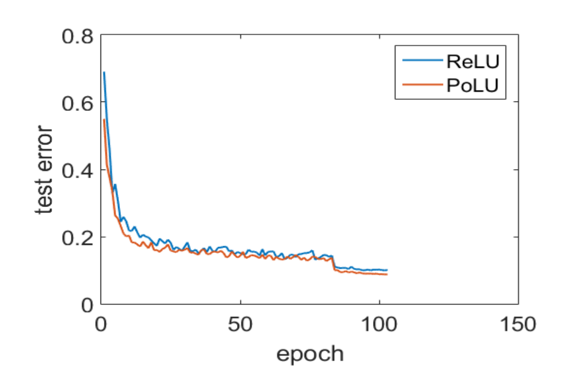


图 3　NIN 使用 ReLU 和 PoLU 的测试误差

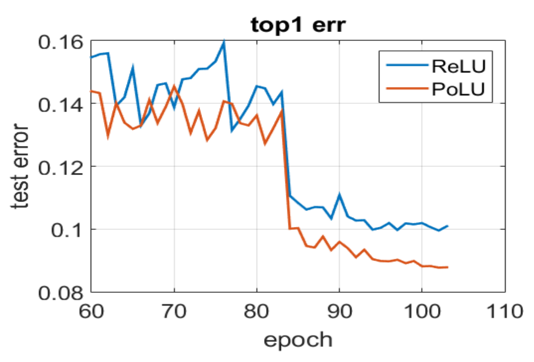


图 4　第 60 个 epoch 之后的测试误差

从图中可以看出，PoLU 激活函数要比 ReLU 激活函数好 1.31%。表 3 对通道独享型 PoLU 和通道共享 PoLU 进行比较。可以看出通道独享型 PoLU 要优于通道共享型 PoLU，并且 PoLU 要优于 PReLU。本文剩余实验全部使用通道独享型 PoLU 和通道独享型 PReLU。

表 3　NIN 使用 ReLU,PReLU 和 PoLU 在 C10 中的对比

方法	Top-1（%）
ReLU[1]复现	10.10
PReLU[3]（通道共享型）	9.43
PReLU[3]（通道独享型）	9.31
PoLU（通道共享型）	8.91
PoLU（通道独享型）	8.79

3.2.3 与其他非线性激活函数比较

本文比较 PoLU 和其他四种激活函数 ReLU、LReLU、ELU 和 PReLU，比较实验在四个数据库进行，分别是 C10、C10+、C100 和 C100+。对于，每一个数据库，当从头开始训练一个新网络模型时，只更改激活函数而保持其他设置不变。使用 ReLU 作为激活函数的 NIN 网络模型在 C10 上和 C100 上的复现 top-1 测试误差分别是 10.10%和 33.02%，这两个复现实验结果都好于文献[10]报道的结果。在 C10+上复现的 ReLU 激活函数实验结果是 8.83%，与文献[10]的 baseline 具有可比性。

表 4 是关于对比实验结果的总结。在所有激活函数中 PoLU 表现最好。在 C10 上，PoLU 提升了 ReLU 的性能从 10.10%到 8.79%；在 C10+上，PoLU 提升了 ReLU 的性能从 8.83%到 7.84%；在 C100 上，PoLU 提升了 ReLU 的性能从 33.02%到 31.73%；在 C100+上，PoLU 提升了 ReLU 的性能从 32.51%到 30.64%。大多数情况下，LReLU，ELU 和 PReLU 表现都比 ReLU 要好，并且 PReLU 在四者表现最好。相比 PReLU，本文的 PoLU 分别在 C10、C10+、C100 和 C100+提升 0.52%、0.13%、0.91%和 0.77%。

表 4　ReLU、LReLU、ELU、PReLU 和 PoLU 的对比实验

方法	C10	C10+	C100	C100+
文献[10]ReLU[1]复现	10.4110.10	8.818.83	35.6833.02	-32.51
LReLU[2]复现	9.53	8.75	33.22	31.95
ELU[3]复现	9.33	8.14	32.73	31.57
PReLU[4]复现	9.31	7.91	32.64	31.41
PoLU(本文)	8.79	7.78	31.73	30.64

注:实验网络 NIN,所用数据库 C10/C10+和 C100/C100+。结果是 top-1 测试误差(%)

3.3　学习到的参数分析

如文献[16]所示，CNN 中更深的层会抓取更多语义信息，如全连接层和最后一个卷积层；然而浅层更类似 Gabor 滤波器对边缘和纹理更敏感，如第一个卷积层。表 5 展示了 NIN 中每一个 PoLU 层学习到的参数。从表 5 中可以看出，越深的 mlpconv 层学到的参数值越小。这表明随着层数增加，激活函数的非线性越大。相比深层，浅层负激活部分包含更多信息，因此学习到的参数值也越大。

表 5　C10 上 NIN 使用通道共享 PoLU 学习到的参数

网络层	Conv1	Cccp1	Cccp2	Conv2	Cccp3	Cccp4	Conv3	Cccp5
参数	0.421	0.454	0.451	0.258	0.262	0.327	0.209	0.237

4　结束语

本文提出新颖的参数化激活函数用于深度 CNNs，称为幂线性单元（power linear unit）。不同于现存的激活函数，本文的 PoLU 可以对负激活部分表现出各层不同的非线性变化。在 CIFAR-10 和 CIFAR-100 的实验结果表明，PoLU 可以在 NIN 网络模型上提升性能。同时，本文展示了负激活部分在不同层有着不同的作用，对于 PoLU 的设计和分析有助于更好的理解深度 CNNs。

参考文献：

[1] Glorot X, Bordes A, Bengio Y. Deep sparse rectifier neural networks [C]// Proc of the 14th International Conference on Artificial Intelligence and Statistics. 2011: 315-323.

[2] Maas A L, Hannun A Y, Ng A Y. Rectifier nonlinearities improve neural network acoustic models [C]// Proc of International Conference on Machine Learning. 2013: 3

[3] He Kaiming, Zhang Xiangyu, Ren Shaoqing, *et al.* Delving deep into rectifiers: surpassing human-level performance on imagenet classification [C]// Proc of IEEE International Conference on Computer Vision. 2015: 1026-1034.

[4] Clevert D A, Unterthiner T, Hochreiter S. Fast and accurate deep network learning by exponential linear units (elus) [C]// Proc of International Conference on Learning Representations. 2016.

[5] Goodfellow I J, Warde-Farley D, Mirza M, *et al.* Maxout networks [EB/OL]. (2013-09-20) . https://arxiv. org/abs/1302. 4389.

[6] Hochreiter S. The vanishing gradient problem during learning recurrent neural nets and problem solutions [J]. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 1998, 6 (2): 107-116.

[7] Nair V, Hinton G E. Rectified linear units improve restricted Boltzmann machines [C]// Proc of the 27th International Conference on Machine Learning. 2010: 807-814.

[8] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks [C]// Advances in Neural Information Processing Systems. 2012: 1097-1105.

[9] He Kaiming, Zhang Xiangyu, Ren Shaoqing, *et al.* Deep residual learning for image recognition [C]// Proc of IEEE Conference on Computer Vision and Pattern Recognition. 2016: 770-778.

[10] Lin Min, Chen Qiang, Yan Shuicheng. Network in network [EB/OL]. (2014-03-04) . https://arxiv. org/abs/1312. 4400.

[11] Huang Gao, Liu Zhuang, Weinberger K Q, *et al.* Densely connected convolutional networks [C]// Proc of IEEE Conference on Computer Vision and Pattern Recognition. 2017: 3.

[12] Krizhevsky A, Hinton G. Learning multiple layers of features from tiny images [D]. Toronto: University of Toronto, 2009.

[13] Huang Gao, Sun Yu, Liu Zhuang, *et al.* Deep networks with stochastic depth [C]// Proc of European Conference on Computer Vision. Cham: Springer, 2016: 646-661.

[14] Hinton G E, Srivastava N, Krizhevsky A, *et al.* Improving neural networks by preventing co-adaptation of feature detectors [EB/OL]. (2012-09-03) . https://arxiv. org/abs/1207. 0580.

[15] Ioffe S, Szegedy C. Batch normalization: accelerating deep network training by reducing internal covariate shift [EB/OL]. (2015-03-20) . https://arxiv. org/abs/1502. 03167.

[16] Zeiler M D, Fergus R. Visualizing and understanding convolutional networks [C]// Proc of European Conference on Computer Vision. Cham: Springer, 2014: 818-833.

[17] Vinyals O, Toshev A, Bengio S, *et al.* Show and tell: a neural image caption generator [C]// Proc of Computer Vision and Pattern Recognition. 2015: 3156-3164.

[18] Ren Shaoqing, He Kaiming, Girshick R, *et al.* Faster R-CNN: towards real-time object detection with region proposal networks [C]// Advances in Neural Information Processing Systems. 2015: 91-99.

[19] Jaderberg M, Simonyan K, Zisserman A. Spatial transformer networks [C]// Advances in Neural Information Processing Systems. 2015: 2017-2025.